領域分割法とその並列化(数値計算アルゴリズムの研究)

AUTHOR(S):

野田, 敏達; 小柳, 義夫

CITATION:

野田, 敏達 ...[et al]. 領域分割法とその並列化(数値計算アルゴリズムの研究). 数理解析研究所講究録 1998, 1040: 196-203

ISSUE DATE:

1998-04

URL:

http://hdl.handle.net/2433/62020

数理解析研究所講究録
1040巻 1998年 196-203

領域分割法とその並列化

野田 敏達 (Bintatsu Noda)　小柳 義夫 (Yoshio Oyanagi)
東京大学大学院 理学系研究科 情報科学専攻

1 はじめに

自然現象は、ポアソン方程式などの偏微分方程式によって定式化されることが多い。これらの方程式を離散化して数値的に解く場合、非常に大きな連立一次方程式を解く必要がある。その解法の一つに1870年Schwarzによって考案された領域分割法がある。これは領域を小さく分割することにより、問題サイズを小さくして解く方法である。近年では、Keyesらの並列化の研究[1]などが行われている。領域分割法は小領域間の境界部分の値を更新しながら反復して解く方法が主流であるが、本研究では小領域内をコレスキー分解を用いて内点を消去し、小領域間の境界の値を前処理付共役勾配法を用いて解く領域分割法を提案する。一辺のサイズが M の矩形領域を解くのに必要なこの方法の計算量は $O(M^{8/3})$ であり、一般的に用いられる前処理付共役勾配法やコレスキー分解などの $O(M^3)$ より小さくてすむ。また、AP1000+に実装することによって、高い並列性を持つことも分かった。

2 問題

$$-\nabla k\nabla u = f \quad \text{in} \quad \mathcal{D} \subset \Re^2, \tag{1}$$

ただし k, f はそれぞれ拡散係数、ソースタームを表わす。また、∇ は

$$\nabla^2 u = \frac{\partial^2}{\partial x^2}u + \frac{\partial^2}{\partial y^2}u$$

という問題を離散化して得られる方程式

$$A\boldsymbol{u} = \boldsymbol{f} \tag{2}$$

を考える。このとき $\boldsymbol{u}$ は求めるべき領域の各格子点の値であり、A は正値対称行列となる。

3 領域分割法

図1のように、領域 Ω を R 個の接っしない小領域 $\Omega_0, \Omega_1, \Omega_2, \cdots, \Omega_{R-1}$ と、それらの小領域に含まれない内部境界を表わす領域 Ω_R の $R+1$ 個の部分領域に分割する。すると、式(2)は次式のように書ける。

$$\begin{pmatrix} A_{00} & O & O & \cdots & A_{0R} \\ O & A_{11} & O & \cdots & A_{1R} \\ O & O & A_{22} & \cdots & A_{2R} \\ \vdots & \vdots & \vdots & & \vdots \\ A_{R0} & A_{R1} & A_{R2} & \cdots & A_{RR} \end{pmatrix} \begin{pmatrix} \boldsymbol{u}_0 \\ \boldsymbol{u}_1 \\ \boldsymbol{u}_2 \\ \vdots \\ \boldsymbol{u}_R \end{pmatrix} = \begin{pmatrix} \boldsymbol{f}_0 \\ \boldsymbol{f}_1 \\ \boldsymbol{f}_2 \\ \vdots \\ \boldsymbol{f}_R \end{pmatrix} \tag{3}$$

$\boldsymbol{u}_r$ は小領域 Ω_r 内の各格子点の値、$\boldsymbol{u}_R$ は内部境界 Ω_R 内の各格子点の値を表す。また、A_{rr} は小領域 Ω_r 内の各格子点の関係、A_{RR} は内部境界 Ω_R 内の各格子点の関係、A_{rR}, A_{Rr} は小領域 Ω_r 内の格子点と内部境界 Ω_R 内の格子点の関係をそれぞれ表している。

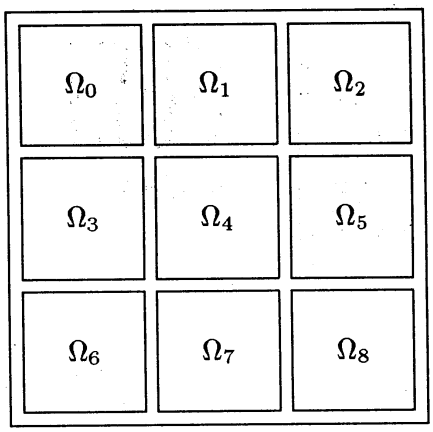

図 1: Domain Decomposition

3.1 Capacitance System

式(3)を書き直すと、

$$A_{rr}\boldsymbol{u}_r + A_{rR}\boldsymbol{u}_R = \boldsymbol{f}_r \tag{4}$$

$$\sum_{r=0}^{R-1} A_{Rr} + A_{RR} = \boldsymbol{f}_R \tag{5}$$

となる。式(4)より得られる

$$\boldsymbol{u}_r = A_{rr}^{-1}(\boldsymbol{f}_r - A_{rR}\boldsymbol{u}_R) \tag{6}$$

を用いて $\boldsymbol{u}_r$ を消去すると、内部境界 $\boldsymbol{u}_R$ だけの方程式

$$C\boldsymbol{u}_R = \boldsymbol{s}_R \tag{7}$$

ただし、

$$C = A_{RR} - \sum_{r=0}^{R-1} A_{Rr}A_{rr}^{-1}A_{rR} \tag{8}$$

$$s_R = \boldsymbol{f}_R - \sum_{r=0}^{R-1} A_{Rr}A_{rr}^{-1}\boldsymbol{f}_r \tag{9}$$

を得ることができる。これを capacitance system と呼ぶ。

3.2 Capacitance System の構造

図2の右図は内部境界を1列とった場合の構造である。これは行列のサイズを非常に小さくするが、図のように構造が複雑となる。それに対し、図3の右図は内部境界を2列とった場合の構造である。この行列のサイズは1列のときの2倍であるが、図のように構造はとても単純になる。

3.3 前処理

本研究では境界を2列とって内点消去した capacitance system を前処理付共役勾配法で解く方法を提案する。この capacitance 行列は非常に単純であるため、前処理にはブロック対角部分コレスキー分解を容易に用いることができる。数式で表わすと、前処理行列を M とすると、

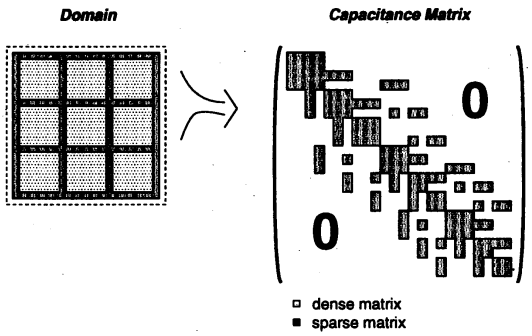

図 2: one line boundaries

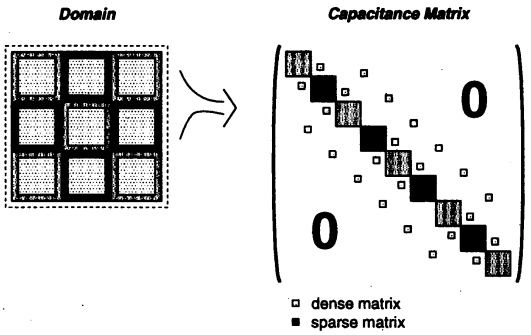

図 3: two lines boundaries

Cholesky Decomposition of A_{rr}	$O(M^3/K)$
making the Capacitance Matrix	$O(M^2)$
solving the Capacitance Matrix by PCG	$O(M^{5/2}K^{1/2})$
backward substitution on each domain	$O(M^3/K)$

If K is $O(M^{1/3})$, the total computational complexity is the minimum $O(M^{8/3})$

表 1: Computational Complexity

$$U^T D U = M = \sum_{r=0}^{R-1} A_{Rr} A_{rr}^{-1} A_{rR} = \sum_{r=0}^{R-1} U_{Rr}^T D_r U_{rR} \tag{10}$$

となる。

3.4 計算量

計算方法をまとめると、まず各小領域から得られる行列をコレスキー分解することによって内部境界だけの方程式 capacitance system を作成する。次に capacitance system を前処理付共役勾配法を用いて解くことにより境界の値を求める。最後に、既に行われているコレスキー分解を利用して、各小領域の値を求める。

表 1は $M \times M$ の問題を $K \times K$ 個の小領域に分割する領域分割法の各段階で必要な計算量である。このうち支配的な計算量は A_{rr} のコレスキー分解の $O(M^3/K)$ と Capacitance System を解くための前処理付共役勾配法の $O(M^{5/2}K^{1/2})$ である。総計算量を最小にする K の値は $O(M^{1/3})$ で、そのときの計算量は $O(M^{8/3})$ となる。

4 並列化手法

前節で、領域分割数は問題サイズの $M^{1/3}$ 程度がよいことが分かった。並列化は領域分割数と PE 数が等しい場合、分割数より PE 数が少ない場合、分割数より PE 数が多い場合という 3 つのパターンが考えられる。

4.1 1 小領域を 1PE に割りあてる場合

小領域の数と PE 数が等しい場合は、1 小領域を 1 つの PE に割りあてることになる。図 4と図 5はそれぞれ境界が 1 列の場合と 2 列の場合のときの、capacitance matrix の作られ方を示している。これから分かるように、2 列の場合は通信なしで capacitance matrix を作ることができることが分かる。

図 6は境界を 2 列とったときの capacitance matrix の構造を示している。前処理としてブロック対角部分コレスキー分解を用いる前処理付共役勾配法によって capacitance system を解く場合、その前処理のた

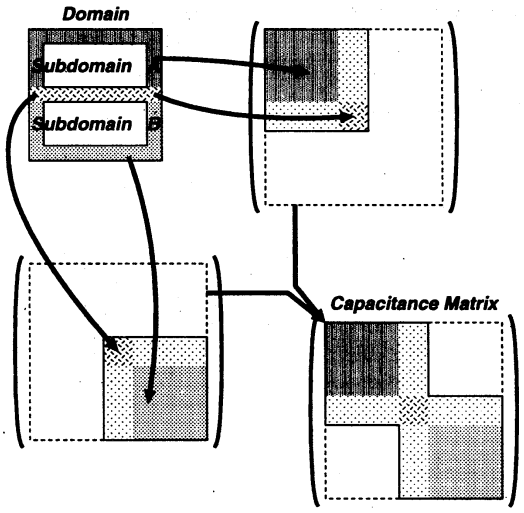

図 4: one line boundaries

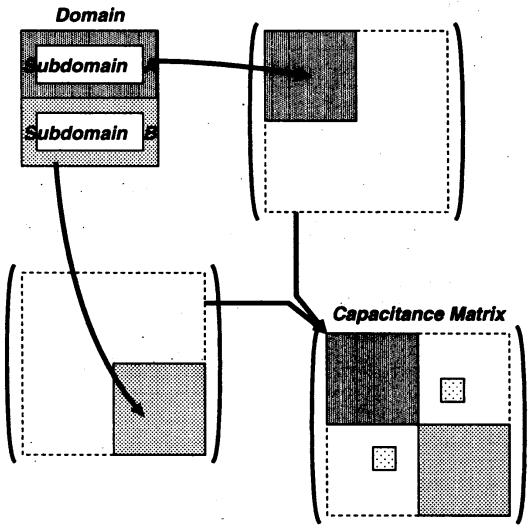

図 5: two lines boundaries

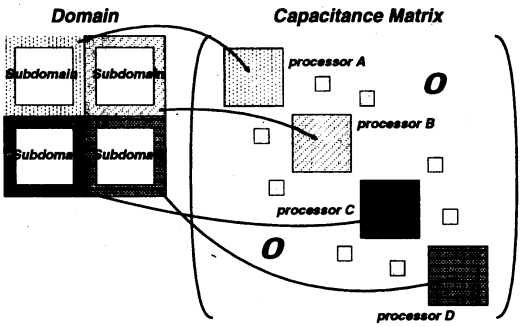

図 6: structure of capacitance system

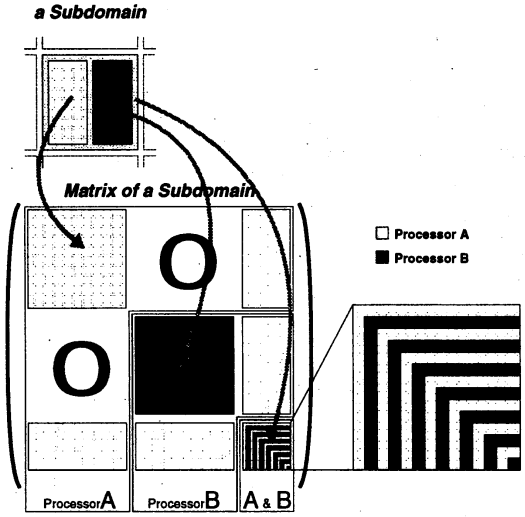

図 7: Cholesky decomposition of a subdomain

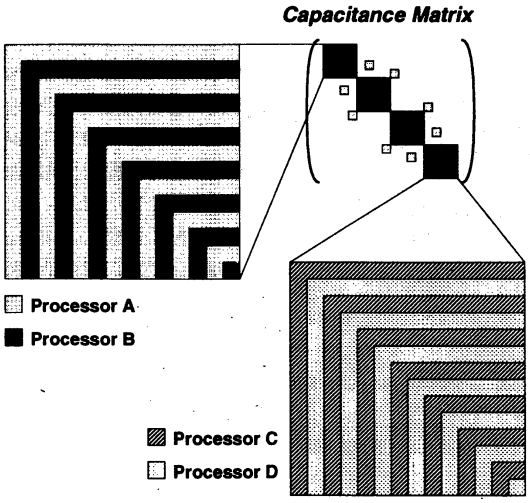

図 8: data constribution of the capacitance matrix

めのコレスキー分解もやはり通信なしですることが可能である。このため必要な通信は前処理付共役勾配法の毎ステップにおける隣接データだけであり、これは行列とベクトルの積に用いられる。

4.2 複数小領域を 1PE に割りあてる場合

小領域の数が PE 数より多い場合は、複数の小領域を 1 つの PE に割りあてることになる。この場合の処理は、1 小領域を 1PE に割りあてる場合とほとんど変わらない。capacitance system を作成する部分、及び最後に小領域を解く部分は全く同じである。capacitance system を解く部分も同一 PE 内での通信をなくすという変更で対応できる。

4.3 1 小領域を複数 PE に割りあてる場合

小領域の数が PE 数より少ない場合、1 つの小領域をいくつかの PE に割りあてることになる。この場合 capacitance system の作成のおいてコレスキー分解を複数の PE をまたいで行う必要がある。しかし、疎行列のコレスキー分解は並列実行に向いた手法ではないため、前節のケースよりもなる。

図 7は 2 つの PE によって分担された小領域にコレスキー分解をどのように適用するか示している。小領域をさらに小々領域に分割することにより、それぞれの小々領域は独立にコレスキー分解を適用することができる。一方少々領域の境界は複数の PE を使用して解く必要があるが、先の 2 つのブロックよりも密となっており、密行列とみなして、ブロックサイクリックデータ分割を行った。この最後のブロックのサイズのオーダーは小領域の 1 辺の長さと等しく、小領域の面積 (= 一辺の長さの 2 乗) と同じサイズである先の 2 つのブロックサイズよりはるかに小さい。よって、この部分の演算量は全体の計算量にそれほど大きな影響を及ぼさない。以上のような方法で capacitance system の作成を行うことにした。

次に capacitance system を解く段階である。図 8はそれぞれ 2 つの PE に分担された 4 つの小領域によって生成される capacitance 行列を示している。これに ブロック対角部分前処理を適用するために、各ブロックにそれぞれコレスキー分解をする必要がある。これらのブロックは密行列なので、ブロックサイクリックデータ分割による並列化を用いた。

最後に、capacitance system を解くことによって境界の値が得られことによって、既に行われているコレスキー分解を利用して、小領域の値を求めることができる。

5 数値実験

$$-\nabla k \nabla u = f \quad \text{in} \quad \mathcal{D} = [0,1] \times [0,1] \tag{11}$$

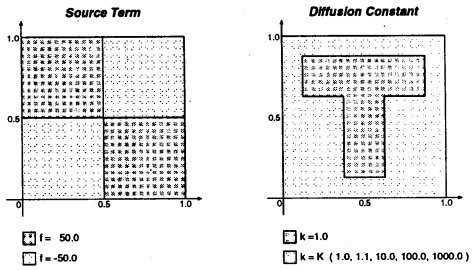

図 9: Problem 1

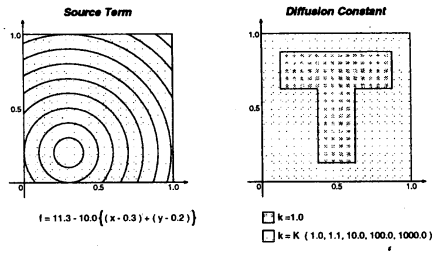

図 10: Problem 2

$k=$	1.0	1.1	10.0	100.0	1000.0
DDM(2×2)	12	43	53	57	61
DDM(4×4)	32	72	81	87	92
DDM(8×8)	45	101	120	131	138
CG	107	267	683	1810	4328
ICCG(1,1)	65	81	94	100	110

表 2: Diffusion Constant 1

$$u = g \quad \text{on} \quad \partial\mathcal{D}. \tag{12}$$

を離散化して得られる方程式を、コレスキー分解、CG 法、ICCG(1,1) 法、DDM(1×1, 2×2, 4×4, 8×8, 16×16) の 5 種類についてそれぞれ比較実験した。

問題は 図 9と図 10の 2 種類を実験した。

5.1 問題サイズと計算量

図 11は、横軸が問題のグリッド数、縦軸が MFlop の問題 1, $k = 1.0$ のグラフである。これより、問題サイズが小さければ分割数が小さい 2×2 がよく、問題サイズが大きくなるに従って、分割数もあげていくとよいことがわかる。この包絡線が最良分割数を用いた領域分割法における計算量となる。これは $O(M^{1/3})$ に分割したとき $O(M^{8/3})$ になることを示している。

5.2 拡散係数と反復回数

表 2はサイズが 64×64 の問題 1 の拡散係数と反復回数を示している。DDM は CG 法よりもはるかに拡散係数の変化に強く、ICCG なみであることが分かる。

5.3 並列化効率

図 12は問題サイズが 64×64 の問題を 1 小領域を 1 プロセッサに割りあてることによって並列に解くのにかかった時間の小領域数倍 (並列台数倍) したものと、同一問題を 1 プロセッサを用いて解いたものの時間の比を表わしている。4 つのグラフは左から、問題 1 $k = 1.0$、問題 1 $k = 100.0$、問題 2 $k = 1.0$、問題 2 $k = 100.0$ の結果である。4×4 で 0.95、8×8 で 0.85 という高い並列化効率を得ることができた。並列台数を増やすと効率が悪くなる理由は、通信が必要な capacitance system を解く時間の割合が増えることと、微妙なロードアンバランスにあると考えられる。

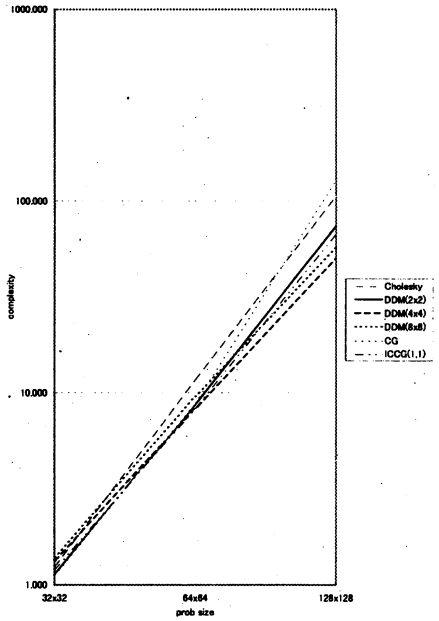

図 11: the complexity of problem 1

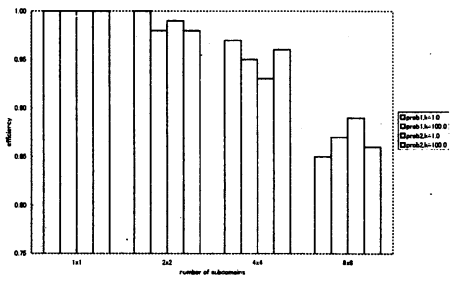

図 12: parallel efficiency

図 13: time to solve the problem

5.4 並列計算時間

図 13は 256PE を使用したときの計算時間である。これも問題サイズが小さいときは分割数が小さいのがよく、問題サイズが大きくなるにしたがって、分割数をあげていけばよいことが分かる。

6 まとめと今後の課題

本論文では、内部境界を二列とり、小領域内の方程式をコレスキー分解、capacitance system(小領域間の方程式) を前処理付共役勾配法で解き、前処理としてブロック対角部分のコレスキー分解を用いる方法を提案した。問題サイズが $M \times M$ の問題を解くとき一般的な方法である 共役勾配法や コレスキー分解を用いた場合、計算量が $O(M^3)$ であるのに対し、領域分割法を用いることによって、$O(M^{8/3})$ で計算できることを実際に並列計算機 AP1000+ に実装することにより確認した。

今回は正方領域の実験しかできなかった。複雑な形状の領域でもよい結果が得られるのかどうか、また、どのような問題を得意とするのか等の検討が今後の課題である。

参考文献

[1] D.E.Keyes, Y.Saad, and D.G.Truhlar. Domain-Based Parallelism and Problem Decomposition Methods in Computational Science and Engineering. *SIAM, Philadelphia, PA, 1995*

[2] Barry Smith, Petter Bjorstad, and William Gropp Domain Decomposition *Cambridge University Press,1996*

[3] Alan George. Nested Dissection of a Regular Finite Element Mesh *SIAM Journal on Numerical Analysis, 10:345-363,1973*